Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по домашнему заданию

по курсу «Анализ Алгоритмов»

на тему: «Графовые модели программ»

Студент группы ИУ7-53Б _____ (Подпись, дата) Дьяченко А. А. (Фамилия И.О.)

Преподаватель _____ (Подпись, дата) Волкова Л. Л. (Фамилия И.О.)

Преподаватель _____ (Подпись, дата) Строганов Д. В.. (Фамилия И.О.)

Москва — 2024 г.

Содержание

1 Введение **3**

1.1 Задание 3

1.2 Графовые модели программы 3

2 Выполнение **4**

2.1 Выбор языка программирования 4

2.2 Код программы 4

3 Графовые модели программы **5**

3.1 Граф управления 5

3.2 Информационный граф 6

3.3 Операционная история 7

3.4 Информационная история 8

1 Введение

1.1 Задание

Описать четырьмя графовыми моделями (граф управления, информационный граф, опрерационная история, информационная история) последовательный алгоритм либо фрагмент алгоритма, содержащий от 15 значащих строк кода и от двух циклов, один из которых является вложенным в другой.

Вариант 6: в качестве реализуемого алгоритма — поиск подстроки в файле.

1.2 Графовые модели программы

Программа представлена в виде графа: набор вершин и множество соединяющих их направленных дуг.

1) **Вершины**: процедуры, циклы, линейный участки, операторы, итерации циклов, срабатывание операторов и т. д.

2) **Дуги**: отражают связь (отношение между вершинами).

Выделяют 2 типа отношений:

1) операционное отношение — по передаче управления;

2) информационное отношение — по передаче данных.

Граф управления — модель, в который **вершины** —операторы, **дуги** — операционные отношения.

Информационный граф — модель, в которой **вершины**: операторы, **дуги** — информационные отношения.

Операционная история — модель, в которой **вершины**: срабатывание операторов, **дуги** — операционные отношения.

Информационная история — модель, в которой **вершины**: срабатывание операторов, **дуги** — информационные отношения.

2 Выполнение

2.1 Выбор языка программирования

Для выполнения домашнего задания был выбран язык C++.

2.2 Код программы

Код программы приведен в листинге 2.1.

Листинг 2.1 – Реализация алгоритма поиска подстроки

```
1     auto startTime = chrono::high_resolution_clock::now(); // (1)
2
3     size_t substringLength = substring.length(); // (2)
4     vector<size_t> positions; // (3)
5     for (size_t i = 0; i < fileContent.length(); ++i) { // (4)
6         if (fileContent.substr(i, substringLength) == substring) {  // (5)
7             positions.push_back(i + shift);  // (6)
8         }
9     }
10    {
11        lock_guard<mutex> lock(mtx);  // (7)
12        ofstream outputFile(outputFilePath, ios::app);  // (8)
13        for (size_t pos : positions) {  // (9)
14            outputFile << pos << "\n";  // (10)
15        }
16        cVars[index].notify_one();  // (11)
17    }
18
19    auto endTime = chrono::high_resolution_clock::now();  // (12)
20    auto elapsedTime = chrono::duration_cast<chrono::milliseconds>(endTime - startTime).count();  // (13)
```

3 Графовые модели программы

3.1 Граф управления

На рисунке 3.1 представлен граф управления.

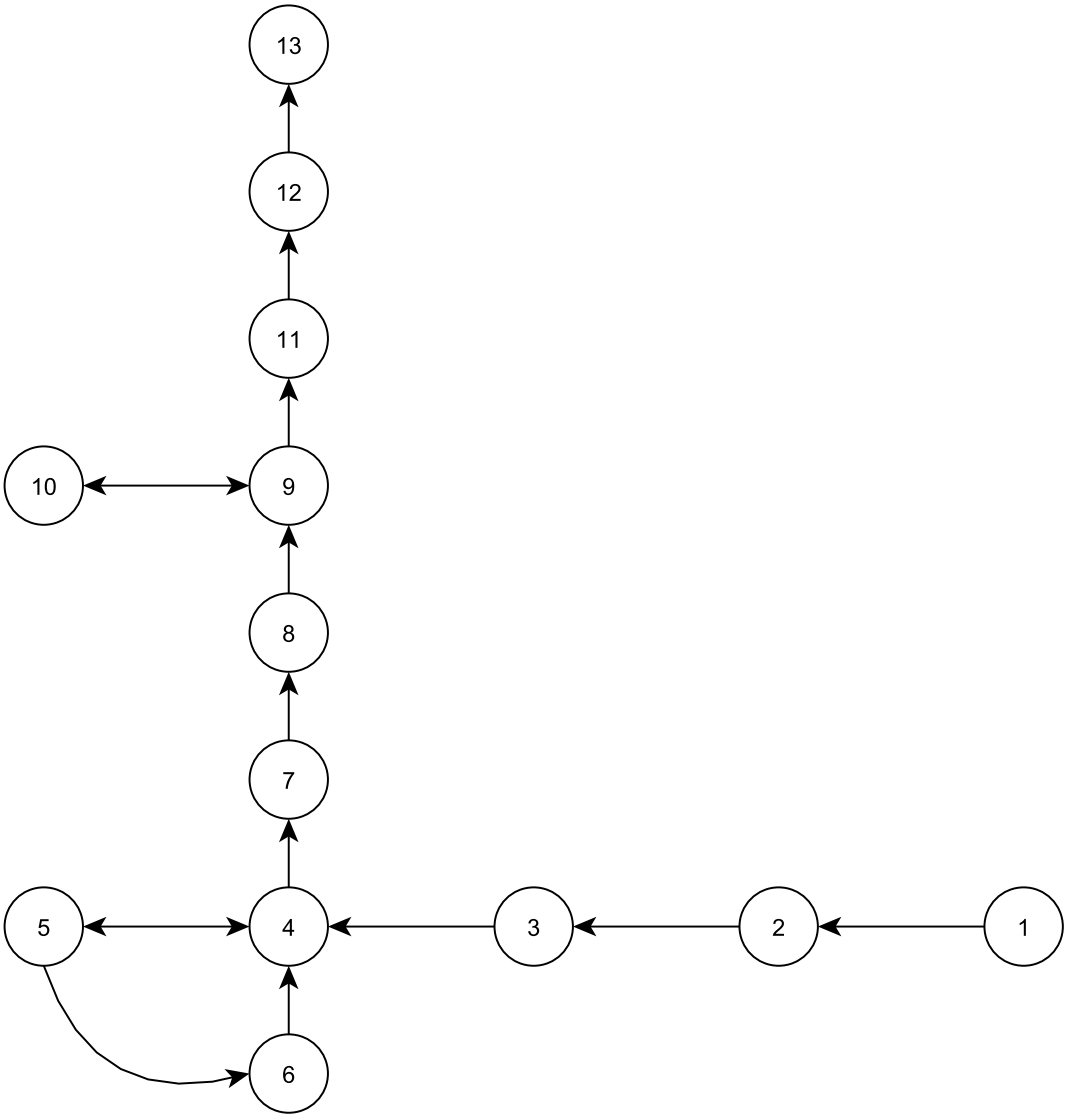

Рисунок 3.1 – Граф управления

3.2 Информационный граф

На рисунке 3.2 представлен информационный граф.

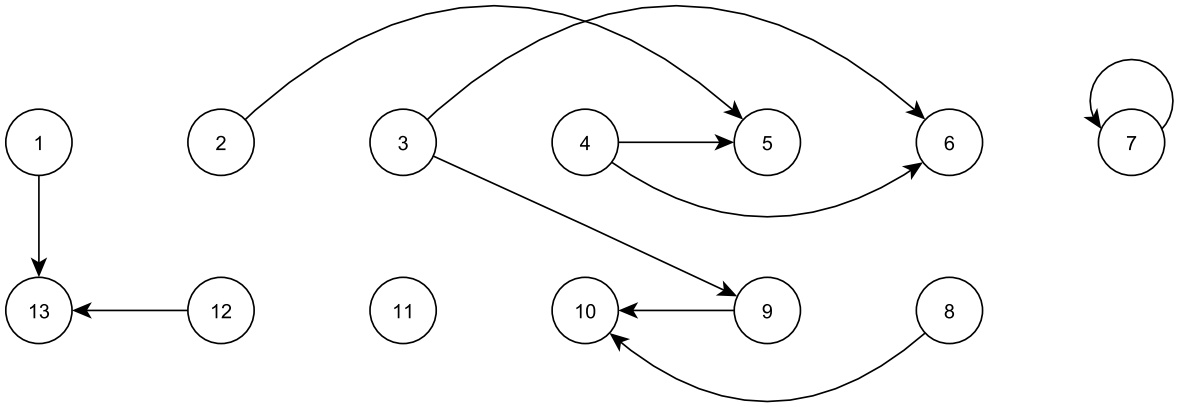

Рисунок 3.2 – Информационный граф

Вершина 7 представляет собой операцию блокировки мьютекса, т.к. файл, куда записываются индексы вхождения подстроки — критическая секция. В информационном графе она связана сама с собой, т.к. операция блокировки мьютекса обращается к этому же мьютексу для проверки его состояния.

Вершина 11 представляет собой установку i-го бинарного семафора массива, который идентифицирует завершение выполнения функции. В графе 3.2 она не связана с другими вершинами потому что используется в управляющем потоке для отслеживания завершения работы каждого потока.

3.3 Операционная история

На рисунке 3.3 представлена операционная история файла, содержащего n символов и m вхождений подстроки.

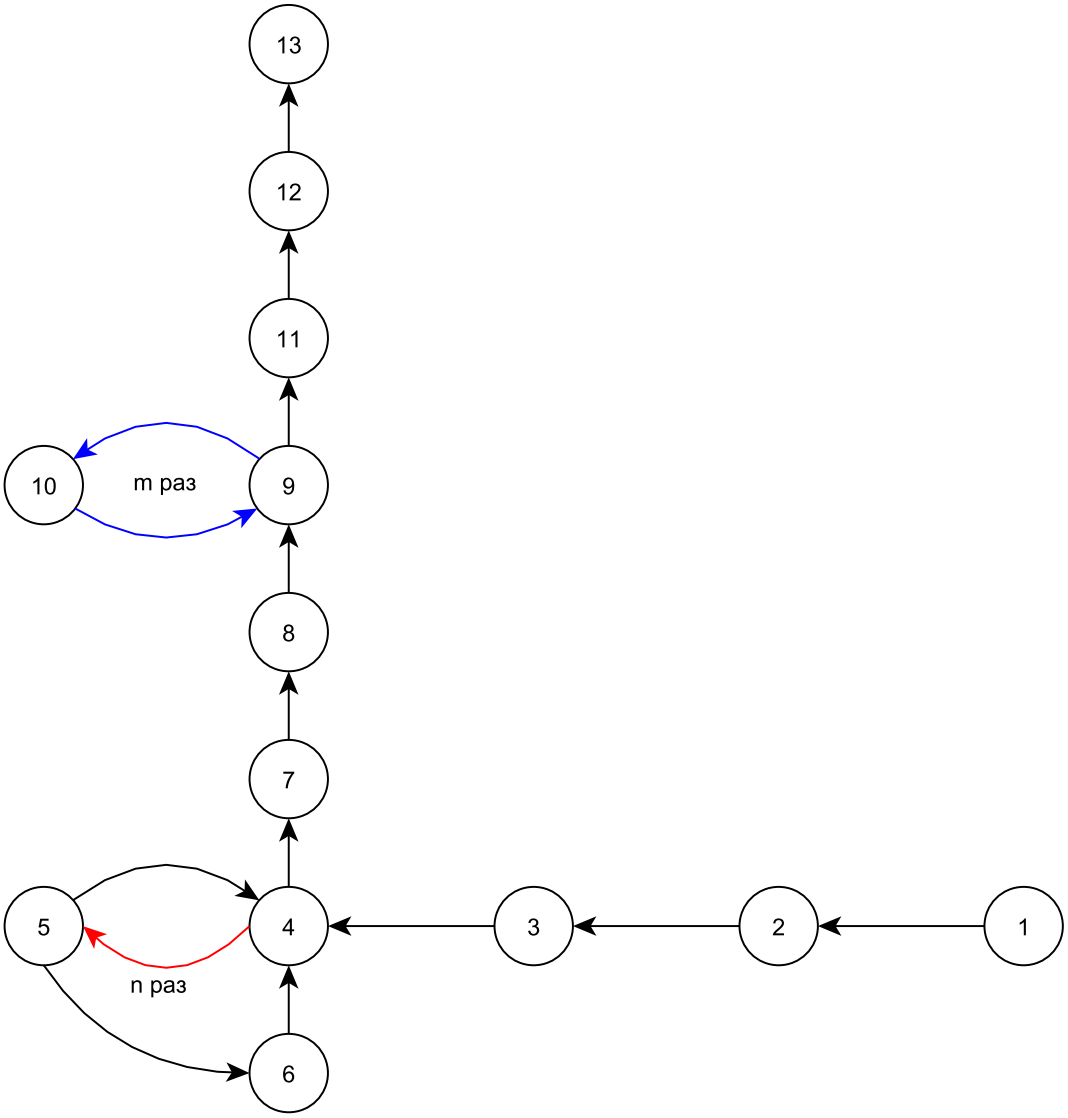

Рисунок 3.3 – Операционная история

3.4 Информационная история

На рисунке 3.4 представлена информационная история файла, содержащего n и m вхождений подстроки.

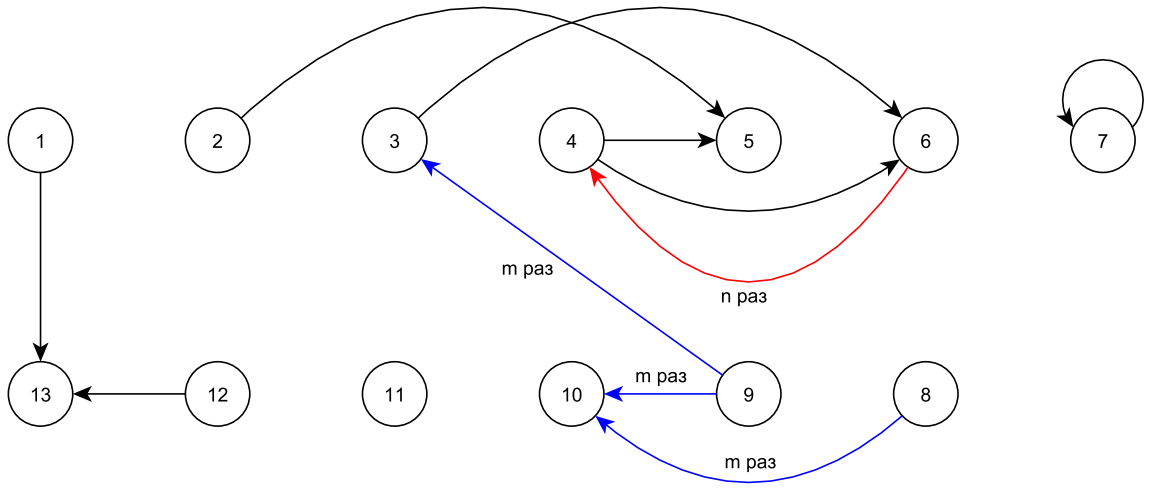

Рисунок 3.4 – Информационная история

Причины, по которым вершина 7 обращается сама к себе, а вершина 11 не связана ни с какой другой, описаны в разделе информационного графа.

Возможность распараллеливания

Распараллеливание алгоритма поиска подстроки в строке (файле) заключается в разделении его на несколько частей и обработке каждой части отдельным потоком.